International Journal of Foundations of Computer Science
© World Scientific Publishing Company

GUARANTEED TERMINATION IN THE VERIFICATION OF LTL PROPERTIES OF NON-LINEAR ROBUST DISCRETE TIME HYBRID SYSTEMS *

WERNER DAMM
GUILHERME PINTO
Department für Informatik, Carl v. Ossietzky Universität Oldenburg,
26111 Oldenburg, Germany
werner.damm@informatik.uni-oldenburg.de

STEFAN RATSCHAN
Institute of Computer Science, Czech Academy of Sciences
Pod Vodarenskou věží 2
182 07 Praha 8
Czech Republic
stefan.ratschan@cs.cas.cz

Received (Day Month Year)
Accepted (Day Month Year)
Communicated by (xxxxxxxxxx)

We present a novel approach to the automatic verification and falsification of LTL requirements of non-linear discrete-time hybrid systems. The verification tool uses an interval-based constraint solver for non-linear robust constraints to compute incrementally refined abstractions. Although the problem is in general undecidable, we prove termination of abstraction refinement based verification and falsification of such properties for the class of *non-linear robust discrete-time hybrid systems.* We argue, that—in industrial practice—safety critical control applications give rise to hybrid systems that are robust. We give first results on the application of this approach to a variant of an aircraft collision avoidance protocol.

Keywords: Non-linear hybrid systems; abstraction refinement; constraint-solving; model-checking; robustness.

1. Introduction

This paper significantly extends previous semi-decidability results for LTL verification of non-linear discrete time hybrid systems with real-valued variables. Even

*This work was partly supported by the German Research Council (DFG) as part of the Transregional Collaborative Research Center "Automatic Verification and Analysis of Complex Systems" (SFB/TR 14 AVACS). See `www.avacs.org` for more information.

though this problem is in general undecidable (by reduction from 2 counter machines), we show, that by exploiting the natural property of *robustness* of real-life hybrid systems, an abstraction-refinement based approach—employing both approximations from above and below—is guaranteed to terminate, either establishing the truth of the requirement, or exhibiting a concrete counterexample, even for non-linear hybrid systems[a]. Although robustness cannot be checked algorithmically, we argue, that every reasonable designed system occurring in practice, is in fact robust, allowing our algorithm to terminate for all practically relevant problem instances.

In contrast, results from Fränzle [15, 16]—also based on robustness arguments—only handle polynomial flows (in a dense time setting). Our approach also improves over other approaches to hybrid systems verification [11, 31, 3, 29, 2, 1, 17, 32] in that termination is guaranteed even for a very rich class of models. Indeed, this is the first paper providing an algorithm with termination proof for full LTL properties for discrete time hybrid systems with non-linear dynamics, which are neither restricted to be controllable, nor to have only finite precision real-valued variables. The only underlying assumption of robustness is close to the one used by Girard and Pappas [17], in that validity may not depend on small perturbations of system variables. In contrast to [17] our notion of robustness applies to the richer setting of non-linear constraints and full LTL verification of the system model.

The presented approach primarily targets safety critical control applications, such as collision avoidance systems, where designs must guarantee separation of traffic agents by safety margins even in the presence of noise and (bounded) inaccuracies of sensors and actuators, and hence *robustness* of designs is a key requirement. Intuitively, for such applications, small variances in measurements or small deviations of actuator settings may not lead to a violation of safety margins between traffic agents. We will give a formal definition of this intuitive concept of robustness, which will be instrumental in establishing termination. Even though robustness cannot be decidable, robustness is "designed into" industrial safety critical applications: Development processes for such applications enforce various measures (such as simulation- or rapid-prototyping based impact analysis of noise, sensor- and actuator inaccuracies) to ensure that deployed designs meet such robustness requirements. Design models used for actual deployment, for example in airborne applications will thus have passed through maturity gates checking that all measures necessary for ensuring robustness have been performed.[b]. The termination argument of our approach thus applies to practically all industrially deployed safety critical applications. For models which do not meet the robustness assumptions the algorithm may fail to terminate, but it will never produce an incorrect answer.

[a]We use the term "hybrid systems" for system models that integrate two different models of execution (state-transition systems for control, and difference equations or differential equations for the plant). The results of this paper apply to the special case of discrete-time hybrid systems.
[b]We have validated this statement with our industrial cooperation partners in the automotive and avionics domain, see www.safetrans-de.org

The approach will be illustrated by an air traffic conflict resolution example [34], where aircrafts follow circular trajectories along opposite directions, leading to a non-linear hybrid system.

As mathematical model we use *discrete time hybrid automata*, which in each time step of fixed duration update a set of real-valued variables as determined by assignments occurring as transition labels, allowing possibly non-linear arithmetic expressions. This subsumes the capability to describe the evolvement of plant variables by difference equations. Transition guards can be non-linear arithmetic constraints. Steps of the automata are assumed to take a fixed time-period (also called cycle-time), intuitively corresponding to the sampling period of the control unit, and determine the new mode and new outputs (corresponding to actuators) based on the sampled inputs (sensors). We allow arbitrary first-order LTL formulas as requirements. Atoms are arithmetic constraints over the real-valued variables of the hybrid automaton, thus allowing to both express response time requirements (such as "*when crash sensor is activated, the airbag will be ignited within 3 cycles*"), stability properties such as "*the aircraft will be maintained at preselected height*", as well as safety properties (such as "*the distance between two aircraft will always be greater than four miles"*).

The decision to base our analysis on discrete time models of hybrid systems is motivated from an application perspective. Industrial design flows for embedded control software entail a transition from continuous time models in early analysis addressing control law design, to discrete time models as a basis for subsequent code generation. Typically, the control-law design is carried out in CASE tools such as Matlab-Simulink in a continuous time setting, validating both stability and safety requirements. Once these key requirements are established, engineers determine sampling rates using standard text book methods (e.g., [22]), and then informally validate the discretized model, typically by simulation. For example, if design engineers have decided to implement a certain control law as a periodic task with periodicity δ, then simulation would be used to "verify" that both stability of the controller as well as safety requirements are maintained in spite of the now limited observability of the plant at the chosen sampling rate δ. The methods of this paper replace this informal validation. They allow to formally prove for models satisfying the robustness assumptions that—even under the limited discrete time visibility of the plant—LTL requirements, and thus both stability properties as well as safety requirements, are guaranteed by the controller.

Related work for discrete time model for hybrid systems focusses on restricting the system model in order to guarantee theoretical decidability for verification by constructing a suitable finite abstraction. In [2, 1] this is done by requiring finite precision values in the observations; in [32] this is achieved by considering only *controllable* systems (where any state may be reached from any state). Recently (at the same time as the conference version of this paper), an abstraction/refinement approach was proposed [17] to solve *safety* verification problems for *linear* discrete time hybrid systems based on robustness assumptions close to ours. In comparison, we consider full LTL verification of non-linear systems. In most cases, the literature on abstraction and refinement for hybrid systems considers a *continuous* time model [11, 30, 18, 31]. Because of this, a direct comparison of the employed algorithms is not possible. We can, nevertheless, observe some differences concerning the way they compute the abstraction.

While, for example, also tools such as Hypertech [19] and Checkmate [10] do support analysis of non-linear systems, with Checkmate offering the highly optimized flow-pipe representation technique, none of these is guaranteed to terminate for robust models when proving temporal properties of hybrid systems. We also note the potential unsafeness of the Checkmate approach in the construction of the abstract transition relation due to rounding errors – in contrast, our constraint solver guarantees, that rounding errors are conservatively over-approximated in refinement steps. In contrast to [11] we are able to prove a termination criterion for abstraction refinement, and can provide safe abstractions for both verifying and falsifying safety properties. The high potential of interval-based evaluation methods for hybrid system verification has already been demonstrated [31, 28, 29] for a more restricted logic, and without termination guarantees.

There are mechanisms for approximating non-linear continuous-time hybrid systems by rectangular automata arbitrarily closely [18]. However, the approximation has to be done manually, and even verifying only rectangular safety properties on the resulting approximation is still an undecidable problem.

The paper is organized as follows: Section 2 elaborates the notion of *robust hybrid systems*, leading to a new notion of *robust satisfaction* and *robust falsification* of LTL properties; Section 3 shows how to approximate robust satisfaction from above and robust falsification from below by exact satisfaction/falsification on finite approximations, and proves, that for any property that is robustly satisfied (falsified) by a (non-linear) robust hybrid system, one can find a finite approximation that establishes this fact; based on this it shows the termination of an abstraction refinement algorithm for the verification and falsification of LTL properties of discrete-time hybrid systems; Section 4 concludes the paper.

2. Robust Hybrid Systems

The authors have substantial experience in analyzing industrial control unit designs for automotive [4, 6, 8, 14] and avionics applications [5, 25], among others in the context of the competence cluster SafeTRANS (*www.safetrans-de.org*), and the Integrated Project SPEEDS funded by the European commission under contract number AIP5-CT-2006-033471. Based on this industrial background, we derive the following observations:

- For any sensor inputs, a combination of filtering, plausibility checking and voting will be used to derive what is often called *validated inputs*.
- This preprocessing will in particular guarantee a minimal separation be-

tween values assumed by validated inputs, in the following sense: assume, that $v \leq 5$ appears as guard of a transition, then altering the guard to $v \leq 5 \pm \varepsilon$ for some ε smaller than a sensor-dependent constant does not change the mode-switching behavior of the system.

- To take into account noise on actuators and un-modeled disturbances, the controller will enforce a safety margin, separating all legal undisturbed runs from forbidden plant regions by some minimal application dependent constant (catering for noise and disturbances). To this end, deviations induced by disturbances and noise on actuators are detected using closed-loop control, and correcting actions to avoid forbidden states are designed to cater for this difference between ideal and measured trajectories.

Designers hence solve the task to guarantee a safety property φ even in the presence of noise on sensors and actuators and un-modeled disturbances. This entails, that the classical notion of satisfiability is in fact too weak. What is called for, is a notion of *robust satisfiability*, which guarantees φ even in the presence of small bounded uncertainties. In the remainder of this section, we will derive a formal definition of such a notion of robust satisfiability.

We use constraints over the current values and the values at the next time step to specify the transition relation. We also use constraints to explicate the predicates observable on the hybrid system, which define the atomic predicates to be used in first-order LTL requirement specifications on our systems. In addition to variables ranging over reals we will use special *mode variables* written in $\mathsf{sans\ serif\ font}$ that range over symbolic constants denoting modes of hybrid systems.

Definition 1.

- *An* arithmetic expression *is a term (in the predicate-logical sense) with function symbols in* $\{+, \times, \hat{}, \sin, \cos, \exp\}$.
- *An* atomic arithmetic state space constraint *is of the form* $e\ r\ c$*, where* e *is an arithmetic expression,* $r \in \{\neq, =, <, >, \leq, \geq\}$ *is a relational operator, and* c *is a real-valued constant.*
- *A* mode constraint *is an expression of the form* $\mathsf{mode} = m$*, where* m *is a symbolic constant and* mode *a mode variable.*
- *A* state space constraint *is a Boolean combination of atomic arithmetic state space constraints and mode constraints.*

For formally modeling discrete time hybrid systems we assume a finite set M of symbolic constants called *modes* and a finite set $X = \{x_1, \ldots, x_k\}$ of *real-valued variables* (in the formal development, we do not further distinguish between sensors, variables, and actuators). We assume that the real-valued variables range over bounded intervals $\{I^{x_1}, \ldots, I^{x_k}\}$. Moreover, for specifying the variable values at the next time step we will denote by X' the primed versions of the variables in X.

Note that in an engineering context one usually knows bounded intervals for the reasonable values of real variables. Our method can also be used to check that these intervals are never left, by formulating a corresponding safety verification problem that shows that no variable ever reaches a value too close to the boundary of its interval.

Definition 2 (discrete time hybrid system) *A discrete time hybrid system S is a tuple $S = (\tau, \pi_0, \pi_1, \dots, \pi_k, \delta)$ where*

- *τ is a disjunction of state space constraints of the form* $\mathsf{mode} = m \wedge guard \wedge \mathsf{mode}' = m' \wedge transitions$ *where*
  - *$m \in M$ and $m' \in M$,*
  - *guard is a conjunction of atomic arithmetic state space constraints that contains only variables in X*
  - *transitions is a conjunction of atomic arithmetic state space constraints that contains only variables in $X \cup X'$.*
- *π_0 is a state space constraint containing only variables in $X \cup \{\mathsf{mode}\}$, restricting the initial valuation, and*
- *$\pi_1, \dots, \pi_k$ are additional state space constraints containing only variables in $X \cup \{\mathsf{mode}\}$, that we will later use in LTL queries (the* observed propositions*), and*
- *δ is the sampling rate in time units, a positive real number.*

Discrete time hybrid systems are sufficiently expressive to express both plant dynamics as well as (possibly hybrid) controllers. Time is modeled implicitly, in that each step corresponds to a fixed unit delay δ, as motivated in the introduction.

Our example is a discretized variant of an aircraft collision avoidance protocol that we took from [19] (originally developed in [34]), exhibiting non-linear dynamics. Two aircraft, flying in a straight line and orthogonal trajectories at the same altitude, initiate a collision avoidance maneuver when the distance between them reaches 8 miles. Both aircraft turn 90 degrees to the right and start a semi-circle trajectory to the left, as shown on the left-hand side of Figure 1, with fixed angular velocities. The linear velocity is also fixed and the same for both aircraft. After completing the semi-circle, they resume their original trajectories.

The continuous time dynamics during the collision avoidance maneuver is given by

$$
\begin{aligned}
\dot{x}_r &= -v_1 + v_2 \cos\phi_r + \omega_1 y_r \\
\dot{y}_r &= v_2 \sin\phi_r - \omega_1 x_r \\
\dot{\phi}_r &= \omega_1 - \omega_2
\end{aligned}
$$

where x_r and y_r are the relative planar coordinates of aircraft two relative to aircraft one, ϕ_r is the angle between the vector speed of aircraft two relative to the vector speed of aircraft one, and v_1 (v_2) and ω_1 (ω_2) are the linear respectively angular velocity of aircraft 1 (aircraft 2), as illustrated by the right-hand side of Figure 1. Please consult [34] for the non-trivial derivation of the system dynamics. For the purpose of this paper we further specialize the dynamics by choosing a constant linear velocity $v_1 = v_2 = 1$ and angular velocities $\omega_1 = 1$ and $\omega_2 = 0.95$. With these instantiations, the model specializes to

$$\begin{aligned} \dot{x_r} &= -1 + \cos\phi_r + y_r \\ \dot{y_r} &= \sin\phi_r - x_r \\ \dot{\phi_r} &= 0.05 \end{aligned}$$

We are interested in analyzing discretized versions of this collision avoidance strategy under a fixed sampling rate δ with respect to the overall safety requirement, that the distance between the two aircraft never becomes smaller than 4 miles. In a typical design flow, this corresponds to the situation, where the time model in Matlab-Simulink is now changed from continuous time to discrete time, in order to analyze the impact of discretization in maintaining the collision avoidance requirement. To this end, we derive a discrete time version using Euler discretization wrt. the chosen sampling rate δ, by replacing each differential equation $\dot{v} = f(v)$ by $v' = v + \delta * f(v)$, yielding

$$\phi_r' = \phi_r + \delta * (0.05) \ \wedge \ x_r' = x_r + \delta * (y_r - 1 + \cos\phi_r) \ \wedge \ y_r' = y_r + \delta * (\sin\phi_r - x_r)$$

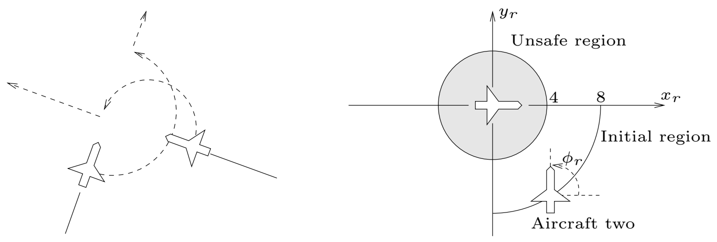

Fig. 1. Air traffic control protocol

where δ is the sampling period of the controller in seconds. We analyze the system for an initial region with $\phi = 1.57 \ \wedge \ x^2 + y^2 = 64 \ \wedge \ x > 0 \ \wedge \ y < 0$, which models an initial configuration before entering the collision avoidance maneuver (before the right-turn of the aircrafts) at a distance of 8 miles, restricted to the cases where the intersection of the trajectories lies ahead of both aircraft

Definition 3.

- *A* valuation *σ is a mapping that assigns a mode in M to the mode variable* mode *and, for each $1 \leq i \leq k$ a real value in I^{x_i} to the variable x_i. We denote the set of all valuations by Σ.*
- *We denote by $[\![\pi]\!]$ the set of all valuations satisfying a state space constraint π, and similarly by $[\![\tau]\!]$ the set of pairs of valuations $\langle\sigma, \sigma'\rangle$ satisfying the*

transition constraint τ, where primed (resp. unprimed) variables are interpreted over σ' (resp. σ).

- *Given a set Γ, we call a tuple $(\rightarrow, Q_0, Q_1, \ldots, Q_k)$, with $\rightarrow \subseteq \Gamma \times \Gamma$, and $Q_0, \ldots, Q_k \subseteq \Gamma$, an* (extended) transition system *over* Γ
- *Given a hybrid system H of the form $(\tau, \pi_0, \pi_1, \ldots, \pi_k, \delta)$ we denote by $[\![H]\!]$ the transition system $([\![\tau]\!], [\![\pi_0]\!], \ldots, [\![\pi_k]\!])$ over Σ.*
- *A* run of a system *$(\tau, \pi_0, \pi_1, \ldots, \pi_k, \delta)$ is a mapping $\theta : \mathbb{N} \rightarrow \Sigma$ such that for all $t \in \mathbb{N}$, $\langle \theta(t), \theta(t+1) \rangle \in [\![\tau]\!]$.*

We use first-order LTL formulas such as $\mathbf{G} \neg x \geq 10$ to formalize requirements on discrete time hybrid systems. Still, the results of this paper hold for any temporal logic using only universal path quantifiers, such as ACTL*. Since steps have a defined duration, real-time constraints can be expressed using the next-time operator. As atoms we allow the observed propositions $\pi_0, \ldots, \pi_k$. We employ standard syntax and semantics of LTL as can be found in various textbooks [13]—the needed adaption to our definition of extended transition system is a trivial exercise. Especially we write $T \models \varphi$ to signify that the extended transition system T satisfies φ.

Note that we do not treat the state space constraint π_0 that specifies the initial states in any special way (e.g., by allowing only runs that start in an initial state). Instead, we encode initial states into the queries by using LTL formulae of the form $\pi_0 \rightarrow \mathbf{G}\varphi$ (i.e., $\neg\pi_0 \vee \mathbf{G}\varphi$).

Robustness of a hybrid system S is defined relative to a temporal specification φ: it requires, that the validity of φ does not depend on small perturbations of S. The formal definition is based on a metric between arithmetic constraints [26]. For S to be *robust wrt.* φ requires the existence of a bound ε, such that if φ holds in S, then it must also hold in any S' defined by constraints that have distance at most ε from the constraints defining S. Intuitively, this entails that avoiding forbidden plant states may not depend on small inaccuracies of sensors or actuators. Indeed, controller designs in which changing a guard of the form $e\ r\ c$ to $e\ r\ (c \pm \varepsilon)$ (mirroring sensor inaccuracy) or changing an actuator setting from $a' = e$ to an assignment $a' = e \pm \varepsilon$ (modeling a small error in actuator settings) causes forbidden states to be reached would not be acceptable (and not "robust", under our formal definition).

We now define these concepts more formally:

Definition 4.

- *The distance between two valuations σ_1, σ_2 is defined by $d(\sigma_1, \sigma_2) \doteq$*
  - ∞, *if* $\sigma_1(\mathsf{mode}) \neq \sigma_2(\mathsf{mode})$, *and*
  - $\max\{d(\sigma_1(x), \sigma_2(x)) \mid x \in X\}$, *where* $d(a,b) \doteq |a - b|$, *otherwise.*
- *The distance between two atomic arithmetic constraints $e\ r\ c$ and $e'\ r'\ c'$ (we assume that all arithmetic constraints have been brought into this form) is defined by $d(e\ r\ c, e'\ r'\ c') \doteq \infty$, if $e \neq e'$ or $r \neq r'$, and $d(c, c')$, otherwise.*

- *The* distance *between two mode constraints* $\mathsf{mode} = m_1$ *and* $\mathsf{mode} = m_2$ *is* ∞ *if* $m_1 \neq m_2$ *and* 0*, otherwise.*
- *The distance between two constraints* ϕ, ϕ' *is defined by* $d(\phi, \phi') \doteq$
  - ∞*, if* ϕ *and* ϕ' *have a different Boolean structure or do not have mode constraints at the same places, and*
  - *the maximum of the distance between two corresponding atomic (arithmetic or mode) constraints, otherwise.*

The key definition of this paper, reported below, captures our intuition that safety properties should be guaranteed even under disturbances, as long as these are bounded by some ε. To this end, we define a *non-standard semantics* of discrete time hybrid systems that allows transitions that miss the original transition predicate only by a slight margin below some ε. For a safety property to be robustly satisfied, there must exist a degree of perturbation $\varepsilon > 0$ such that the safety property is true in all ε-perturbed systems.

We start with defining the notion of an ε-perturbed solution set by stating a condition when x is allowed to be in such a solution set, and a condition when x is allowed to be not in such a solution set. In both cases we require the existence of some perturbations establishing the corresponding fact.

Definition 5. *A set* P *is an* ε*-perturbed solution set of a constraint* ϕ *iff*

- *for every* $x \in P$*, there is a constraint* ϕ^* *with* $d(\phi, \phi^*) \leq \varepsilon$ *and an* x^* *with* $d(x, x^*) \leq \varepsilon$ *such that* $x^* \models \phi^*$*, and*
- *for every* $x \notin P$*, there is a constraint* ϕ^* *with* $d(\phi, \phi^*) \leq \varepsilon$ *and an* x^* *with* $d(x, x^*) \leq \varepsilon$ *such that* $x^* \not\models \phi^*$*.*

In each item of this definition we employ two types of perturbations: a syntactic perturbation of the constraint and a semantic perturbation in the solution space. The motivation for the use of syntactic perturbation lies in the fact that two systems that show the same behavior when corresponding exactly to their model, might show radically different behavior under perturbations. This can be seen, for example, on the constraints $0 = 0$ and $0 \leq 1$ which have the same solution set but behave radically different when perturbing the zero on the left-hand sides of these constraint. The motivation for the use of semantic perturbation lies in the fact that we also want to model drift that external effects might introduce into the system. Moreover, this second type of perturbation also simplifies the proofs in the rest of the paper significantly.

Probably the reason why some authors only consider one of these two types of perturbations (e.g., only semantic [17]) is the fact that in the context of robustness considerations both perturbations behave differently only for special cases (e.g., the constraint $0 = 0$, whose solution set vanishes under syntactic perturbations, but stays the same under semantic perturbations).

Definition 5 extends to hybrid systems as follows:

Definition 6. *A transition system $(\rightarrow, Q_0, Q_1, \ldots, Q_k)$ is an ε-perturbed manifestation of a hybrid system $(\tau, \pi_0, \pi_1, \ldots, \pi_k)$ iff $\rightarrow$ is an ε-perturbed solution set of τ, and for each $i \in \{0, \ldots, k\}$, Q_i is an ε-perturbed solution of π_i.*

This allows us to define robustness of a hybrid system relative to a temporal specification.

Definition 7. *An LTL formula φ is* satisfied *by a hybrid system S with robustness ε ($S \models_\varepsilon \varphi$) iff for all ε-perturbed manifestations T of S, $T \models \varphi$. An LTL formula φ is* robustly satisfied *by S ($S \Vdash \varphi$) iff there is an $\varepsilon > 0$ such that $S \models_\varepsilon \varphi$.*

For example, a system that starts in state $x = 0$ and evolves according to the transition constraint $x' = x$, satisfies the LTL formula $\mathbf{G}\neg x \geq 1$, but does *not* robustly satisfy it, because any transition constraint of the form $x' = x + \varepsilon$, with $\varepsilon > 0$, will eventually violate the constraint $\neg x \geq 1$. On the other hand, a system that evolves according to the transition constraint $x' = x - 1$, robustly satisfies this LTL formula.

Definition 8. *An LTL formula φ is* falsified *by a hybrid system S with robustness ε iff for all ε-perturbed manifestations T of S, $T \not\models \varphi$. An LTL formula φ is* robustly falsified *by S iff there is an $\varepsilon > 0$ such that φ is falsified by S with robustness ε.*

For example, a system that starts in a state fulfilling $0 \leq x \leq 1$, and evolves according to the transition constraint $x \leq x' \wedge x' \leq x + 1/10$ robustly falsifies the LTL formula $\mathbf{G}\neg x \geq 10$.

It is worthwhile to reflect on the nature of the two simple examples of non-robust satisfaction respectively falsification, by setting them into an application perspective. Both examples would indeed fall short of being accepted in industrial settings, because they fail to compensate the possible effect of disturbances and noise. Consider, for example, a speed monitoring system for a train. The example following Definition 7 would correspond to a design, in which the current controlling the engine is set to maintain the targeted speed, without checking that the actual speed of the train is meeting the targeted speed. Unmodelled disturbances, such as the slope of the track, can easily cause the actual speed to grow beyond the critical maximal speed (potentially causing derailing of the train): just consider the situation, where the train is running through a segment of the track with a constant negative slope, adding in each step an ε unintended increment to the current speed. It is exactly for this reason that no control-engineer would rely on open-loop control for such applications; indeed, by providing a feedback-loop compensating for a possible difference between the intended set point (where x is to remain unchanged forever), and the real value in the physical system, the unintended growth of x would be detected and compensated. Also the example illustrating Definition 8 is an open-loop system.

3. Effective Construction of Finite Abstractions with Bounded Imprecision

Our approach follows the abstraction refinement paradigm. In contrast to previous approaches, we are able to prove termination of the refinement loop under the assumption, that the analyzed model is robust. In this section we introduce the key instrument—a bound on the degree of imprecision introduced by abstraction. By proving that incremental refinements make the degree of imprecision converge to zero, any desired degree of precision can be reached. We also show in this section, that such abstractions can be efficiently computed even for non-linear hybrid systems, using interval arithmetic. The last part of this section puts all pieces together in defining an algorithm for proving robust first-order LTL properties and proving its termination.

From now on, we fix a discrete time hybrid system $S = (\tau, \pi_0, \pi_1, \ldots, \pi_r, \delta)$, and a LTL requirement φ on S over the atoms $\pi_0, \ldots, \pi_r$. For the rest of the development, it will be convenient to assume, that negations occur only in literals, and that all atoms appear under the scope of a negation (this can easily be achieved by adapting the relational operators in arithmetic constraints). This allows us to over-approximate the behavior of a hybrid system by over-approximating the observed propositions $\pi_0, \ldots, \pi_r$ in the same direction as the transition relation τ, allowing more uniformity in the algorithms and proofs. So, by over-approximating the solution set of π_0 and π_1 in a query of the form $\mathbf{G}(\neg\pi_0 \vee F\neg\pi_1)$, we under-approximate the literals $\neg\pi_0$ and $\neg\pi_1$.

We use abstractions that approximate the behavior of the original system, and then we measure the approximation error introduced by these abstractions.

Definition 9. *Let T be a transition system over Γ of the form $(\rightarrow, Q_0, \ldots, Q_r)$ and let T' be a transition system of the form $(\rightarrow', Q'_0, \ldots, Q'_r)$ over Γ'. Then T'* abstracts *T ($T' \succeq T$, $T \preceq T'$) iff there is a function H (the* abstraction function*) such that*

- *for all $i \in \{0, \ldots, r\}$, for all $s \in Q_i$, $H(s) \in Q'_i$, and*
- *for all s, s_1 with $s \rightarrow s_1$, $H(s) \rightarrow H(s_1)$.*

Clearly, for transition systems T and T' such that $T \subseteq T'$ (with $\subseteq$ defined element-wise) $T \preceq T'$ (e.g., using the identity abstraction function). Moreover, the relation $\preceq$ is transitive.

The abstraction relation implies the existence of corresponding runs:

Lemma 10. *For transition systems T and T' such that T' abstracts T, for every run θ of T there is a run θ' of T' such that for all $i \in \{1, \ldots, r\}$, for all $t \in \mathbb{N}$, $\theta(t) \in Q_i$ implies $\theta'(t) \in Q'_i$*

The proof is easy by element-wise application of the abstraction function. Moreover, due to Theorem 5.6. in Clarke et. al. [12] we have:

Lemma 11. *For every transition system T and T', for every LTL formula φ, if $T \preceq T'$ then $T' \models \varphi$ implies $T \models \varphi$.*

So, for showing satisfiability we will try to construct transition systems that abstract the original system (i.e., an over-approximation), and for falsification transition systems that are abstracted by the original system (i.e., an under-approximation).

3.1. *Over-Approximating Abstraction*

We use predicate abstraction, tuned to our application domain of hybrid systems. In this framework, the abstract state space is given by a finite set of first-order predicates P, which jointly cover the concrete state space, that is for all σ in Σ there is a $p \in P$ such that $\sigma \in [\![p]\!]$ (note that in contrast to some approaches in software model checking, here the abstract states are formed by the predicates themselves and *not* Boolean combinations of them).

Different approaches for finding P have been discussed in the literature. For example, an initial set of predicates can be derived from transition guards and atomic formulas in the specification logic [7]; or a certain class of predicates, such as convex polyhedra [10], or hyper-rectangles [29] can be used.

For a given finite set of predicates P, we construct an abstraction $\overline{\alpha}_P(S)$ (*natural abstraction*) of $[\![S]\!]$. It is a transition system whose transition relation is the set of all $\langle p, p' \rangle$ for which there is a pair $\langle \sigma, \sigma' \rangle \in [\![\tau]\!]$ such that $\sigma \models p$ and $\sigma' \models p'$. The set of initial states, and the observed propositions are defined canonically as the set of all p for which there is a $\sigma \in [\![\pi_i]\!]$ such that $\sigma \models p$. We get an abstraction function between the concrete infinite state transition system $[\![S]\!]$ and $\overline{\alpha}_P(S)$ by assigning to each state space element σ of S a predicate p such that $\sigma \models p$. Due to Lemma 11 for all first-order LTL formulas φ, $\overline{\alpha}_P(S) \models \varphi$ implies $S \models \varphi$.

Note that here the abstract transition relation also might contain self-loops, that is, transitions from a predicate to itself, if the transition relation τ specifies transitions between between two elements satisfying the same abstract state. If all trajectories eventually leave a certain region, then the corresponding self-loops will be removed as soon as the abstraction is fine enough. This allows the method to prove progress properties.

We now introduce the notion of the *diameter* of a predicate abstraction to later measure the degree of imprecision introduced by an abstraction.

Definition 12. *The diameter $diam(p)$ of a predicate $p \in P$ is defined as the supremum of $\{d(\sigma, \sigma^*) \mid \sigma \in [\![p]\!], \sigma^* \in [\![p]\!]\}$. The diameter $diam(P)$ of a predicate abstraction over P is defined as the maximal diameter of a predicate in P.*

To bound the degree of imprecision of abstraction we will ensure that for every $\varepsilon > 0$ the abstraction eventually only represents a ε-perturbation of S. Hence, the query will eventually be proven on the abstraction. Since it is hard to compare the discrete and finite abstraction with the transition system denoted by S, we will measure these perturbations not from the abstraction $\overline{\alpha}_P(S)$ directly, but from the following continuous over-approximation:

Definition 13. *A transition system* $(\to, Q_0, Q_1, \ldots, Q_r)$ *over a set of predicates* P represents *the transition system*

$$\gamma(\to, Q_0, \ldots, Q_r) \doteq (\gamma(\to), \gamma(Q_0), \gamma(Q_1), \ldots, \gamma(Q_r)),$$

where $\gamma(R) = \bigcup_{p \in R} [\![p]\!]$.

It is not hard to prove that, using an abstraction function that assigns to each $p \in P$, an element of $[\![p]\!]$, for every transition system T over P, $\gamma(T)$ abstracts T. Hence any query φ that is satisfied by $\gamma(T)$ is also satisfied by T, and in particular $\gamma(\overline{\alpha}_P(S)) \models \varphi$ implies that model checking the abstraction will succeed, that is $\overline{\alpha}_P(S) \models \varphi$.

So we are left with the task of showing that $\gamma(\overline{\alpha}_P(S))$ will be sufficiently close to $[\![S]\!]$. For this, given a constraint ϕ, let $\overline{[\![\phi]\!]}_\varepsilon$ be the set of all x for which there is a ϕ^* with $d(\phi, \phi^*) \leq \varepsilon$ and an x^* with $d(x, x^*) \leq \varepsilon$ such that $x^* \models \phi^*$. Clearly this forms the maximal element of the ε-perturbed solution sets of a constraint ϕ wrt. the partial order $\subseteq$. Extending this to every constraint defining a hybrid system, we denote the transition system given by the resulting maximal elements by $\overline{[\![S]\!]}_\varepsilon$, and we have:

Theorem 14. $\overline{[\![S]\!]}_{diam(P)}$ *abstracts* $\gamma(\overline{\alpha}_P(S))$.

We do not include the proof since it can be adapted from the proof of Theorem 17 below. We can conclude that an abstraction $\overline{\alpha}_P(S)$ only introduces bounded perturbations since it can be sandwiched between the exact system $[\![S]\!]$ and its perturbed version $\overline{[\![S]\!]}_{diam(P)}$ due to the fact $[\![S]\!] \preceq \overline{\alpha}_P(S) \preceq \gamma(\overline{\alpha}_P(S)) \preceq \overline{[\![S]\!]}_{diam(P)}$.

The natural abstraction can be constructed effectively, if we do not allow the transcendental function symbols $\sin, \cos, \exp$ in our constraints. For this we decide [33] satisfiability of $p(x_1, \ldots, x_k) \wedge \tau(x_1, \ldots, x, x'_1, \ldots, x'_k) \wedge p(x'_1, \ldots, x'_k)$ for defining the abstract transition relation, respectively $p \wedge \pi_i$ for determining the set of initial states and observed propositions. However, due to the huge complexity of the corresponding decision procedure [9], this approach is not viable in practice.

Consider thus a predicate abstraction of S, where each predicate is of the form $\mathsf{mode} = m \wedge B$, where B is a so-called *box* of the form $\bigwedge_{i \in \{1,\ldots,k\}} c_{i,l} \leq x_i \leq c_{i,u}$. We will usually write such *box predicates* as pairs $\langle m, B \rangle$.

In this case the computational effort in constructing the abstract transition relation can be drastically reduced by using interval arithmetic based tests instead of full decision procedures (the cost of a single test reduces from non-elementary in the number of variables to linear in the expression size). Moreover, this does not restrict the allowed function symbols to addition and multiplication. In this approach, transitions from box p to box p' are only added, if they cannot be excluded by interval arithmetic. We thus further abstract from the concrete transition behavior.

More specifically, we evaluate terms over boxes by extending all function symbols f to corresponding functions f^I over intervals. For example, the arithmetic expression $xy + 1$ for a box that restricts x to $[-1, 1]$, and y to $[1, 2]$, evaluates to $[-1, 1][1, 2] +^I [1, 1] = [-2, 2] +^I [1, 1] = [-1, 3]$. Given an arithmetic expression e and a box B we denote by $I(e)(B)$ the interval evaluation of e over B.

The properties of interval evaluation of terms have been widely studied [24, 21]. Here we use a version that is extended to constraints. Using the Booleans $\{\mathbf{F}, \mathbf{T}\}$ with the order $\mathbf{F} < \mathbf{T}$ one can form Boolean intervals, which allows us to extend relations and connectives to intervals in a similar way as above. Hence we can evaluate Boolean combinations of equalities and inequalities over intervals. The formalization of this is a trivial exercise. For example, the evaluation of the constraint $2x \geq 0 \vee x - 2 \geq 0$ over a box restricting x to $[1, 3]$ yields $[2, 2][1, 3] \geq^I [0, 0] \vee^I [1, 3] - [2, 2] \geq^I [0, 0] = [2, 6] \geq^I [0, 0] \vee^I [-1, 1] \geq [0, 0] = [\mathbf{T}, \mathbf{T}] \vee^I [\mathbf{F}, \mathbf{T}] = [\mathbf{T}, \mathbf{T}]$. One can easily incorporate mode constraints by evaluating a constraint of the form $\mathsf{mode} = m_0$ over a box predicate $\langle m, B\rangle$ to $[\mathbf{T}, \mathbf{T}]$ iff $m = m_0$ and to $[\mathbf{F}, \mathbf{F}]$, otherwise.

Whenever such an evaluation yields an interval $[\mathbf{F}, \mathbf{F}]$ we know that the corresponding constraint cannot hold. So we get a conservatively over-approximated satisfaction relation $\models_I$ such that $\langle m, B\rangle \models_I \phi$ iff $\mathbf{T}$ is in the interval evaluation of ϕ on $\langle m, B\rangle$. Hence $\langle m, B\rangle \not\models_I \phi$ tells us that ϕ cannot be satisfied by mode m and an element of B, whereas $\langle m, B\rangle \models_I \phi$ does not tell us anything since in the case where interval evaluation is $\{\mathbf{F}, \mathbf{T}\}$, the element $\mathbf{T}$ might be spurious due to over-approximation.

Now, by using the over-approximated satisfiability $\models_I$ we get another abstraction $\overline{\alpha}^I_P(S)$ (the *interval abstraction*) for a given set of box predicates P. Since $\models_I$ over-approximates $\models$, also $\overline{\alpha}^I_P(S) \supseteq \overline{\alpha}_P(S)$, and hence $\overline{\alpha}^I_P(S) \succeq \overline{\alpha}_P(S)$. However, we again have to establish that this abstraction only introduces bounded over-approximation:

We start with providing bounds for interval evaluation. By its Lipschitz continuity (e.g., Theorem 2.1.1 in Neumaier's book [24]), it is easy to derive the following convergence result for interval evaluation of terms:

Lemma 15. *For every arithmetic expression e with function symbols in the set $\{+, *, \hat{}, \exp, \sin, \cos\}$, denoting a function $[\![e]\!]$ and box B there is a function $E : \mathbb{R}^+ \to \mathbb{R}^+$ such that $\lim_{x\to 0} E(x) = 0$, and for every box B' with $[\![B']\!] \subseteq [\![B]\!]$, for all $y \in I(e)(B')$, there is an $x \in [\![B']\!]$ such that $d([\![e]\!](x), y) \leq E(diam(B'))$.*

Now we can bound the approximation of interval satisfaction on constraints:

Lemma 16. *For every constraint ϕ, mode m and box B there is a function $E : \mathbb{R}^+ \to \mathbb{R}^+$ with $\lim_{x\to 0} E(x) = 0$, such that for every box B' with $[\![B']\!] \subseteq [\![B]\!]$, $\langle m, B'\rangle \models_I \phi$ implies that there is a ϕ^* with $d(\phi, \phi^*) \leq E(diam(B'))$ and an $x \in [\![B']\!]$ such that $\langle m, x\rangle \models \phi^*$.*

Proof. Let ϕ, m, and B be arbitrary but fixed. Let us first assume that ϕ is an atomic arithmetic constraint of the form $e \;\; \geq \;\; c$. Choose E as provided by Lemma 15, let B' be arbitrary, but fixed, and assume $B' \models_I e \geq c$. In the case when $I(\phi)(B') = \{\mathbf{T}\}$, the rest is trivial. In the case when $I(\phi)(B') = \{\mathbf{F}, \mathbf{T}\}$, $c \in I(e)(B')$ and we can choose $y = c$ in Lemma 15, which provides a corresponding $x \in [\![B']\!]$ such that $d([\![e]\!](x), c) \leq E(diam(B'))$. This implies $[\![e]\!](x) \geq c - E(diam(B'))$. Choosing ϕ^* as $e \geq c - E(diam(B'))$ clearly $d(\phi, \phi^*) \leq E(diam(B'))$ and $\langle m, x \rangle \models \phi^*$.

The case of other atomic constraints with different relation symbols are similar, and the case of mode constraints is trivial. In the case where ϕ is non-atomic we can choose E as the maximum of the E's of its atomic sub-constraints and choose ϕ^* by taking for each atomic constraint the corresponding constraint constructed above. □

Note that in practice—in order to ensure efficiency—interval arithmetic is usually implemented using floating point arithmetic. In that case, all the necessary operations are rounded outwards. So, differently from other methods, we preserve correctness also under the presence of rounding. In principle, for achieving a tight enough over-approximation, one would need to adjust the precision of the used floating-point representation to the level of robustness of the given verification problem. However, experience has shown that for cases arising in practice the usual machine floats suffice.

Finally we can establish an analogous result to Theorem 14:

Theorem 17. *There is a function* $E : \mathbb{R}^+ \to \mathbb{R}^+$ *with* $\lim_{x \to 0} E(x) = 0$*, such that given a set of box predicates* P*,* $\overline{[\![S]\!]}_{E(diam(P))}$ *abstracts* $\gamma(\overline{\alpha}^I_P(S))$*.*

Proof. Let $\overline{\alpha}^I_P(S)$ be of the form $(\to, Q_1, \ldots, Q_r)$. Let E_τ be the function given by Lemma 16 for the transition constraint τ of S and the box $I^{x_1} \times \ldots \times I^{x_k}$ bounding the state space, and $E_{\pi_1}, \ldots, E_{\pi_k}$ be the functions given by Lemma 16 for the state space constraints $\pi_1, \ldots, \pi_k$ of S and the bound of the state space $I^{x_1} \times \ldots \times I^{x_k}$. Let $E(x) \doteq \max\{x, E_\tau(x), E_{\pi_1}(x), \ldots, E_{\pi_r}(x)\}$. We prove that $\overline{[\![S]\!]}_{E(diam(P))} \supseteq \gamma(\overline{\alpha}^I_P(S))$, with $\supseteq$ interpreted element-wise which implies the theorem.

- For an arbitrary, but fixed $i \in \{0, \ldots, r\}$, for proving that every element σ of $\gamma(Q_i)$ is in the corresponding element of $\overline{[\![S]\!]}_{E(diam(P))}$, we prove that it is an element of an $E(diam(P))$-perturbed solution set of the corresponding state space constraint π_i. Observe that, by Definition 13, there is a corresponding element p of Q_i such that $\sigma \models p$. By definition of interval abstraction, $p \models_I \pi_i$. So, by Lemma 16, there is a π_i^* with $d(\pi_i, \pi_i^*) \leq E_{\pi_i}(diam(P))$, and σ^* with $\sigma^* \models \pi_i^*$. Since $diam(p) \leq diam(P)$, also $d(\sigma, \sigma^*) \leq diam(P)$. So, by Definition 5, every element σ of $\gamma(\pi_i)$ satisfies the transition constraint up to $E(diam(P))$.
- For proving that every element $\langle \sigma, \sigma' \rangle$ of $\gamma(\to)$ is in the corresponding element of $\overline{[\![S]\!]}_{E(diam(P))}$, we have to prove that it is an element of an

$E(diam(P))$-perturbed solution set of the transition constraint τ. Observe that by Definition 13 there is a corresponding transition $\langle p, p'\rangle$ in $\rightarrow$ such that $\sigma \models p$, and $\sigma' \models p'$. By definition of interval abstraction, $\langle p, p'\rangle \models_I \tau$. So, by Lemma 16, there is a constraint τ^* with $d(\tau, \tau^*) \leq E_\tau(diam(P))$, and $\langle \sigma^*, \sigma'^*\rangle$ with $\langle \sigma^*, {\sigma'}^*\rangle \models \tau^*$. Since $diam(p) \leq diam(P)$ and $diam(p') \leq diam(P)$, also $d(\sigma, \sigma^*) \leq diam(P)$ and $d(\sigma', {\sigma'}^*) \leq diam(P)$. So, by Definition 5, every element $\langle \sigma, \sigma'\rangle$ of $\gamma(\rightarrow)$ satisfies the transition constraint up to $E(diam(P))$. □

So also the abstraction $\overline{\alpha}^I_P(S)$ only introduces bounded perturbations since it can be sandwiched between the exact system $[\![S]\!]$ and its maximally $E(diam(P))$-perturbed version $\overline{[\![S]\!]}_{E(diam(P))}$ due to the result $[\![S]\!] \preceq \overline{\alpha}^I_P(S) \preceq \gamma(\overline{\alpha}^I_P(S)) \preceq \overline{[\![S]\!]}_{E(diam(P))}$. By decreasing the diameter of P, the precision of the abstraction can be arbitrarily increased. We will use these results in the development of an algorithm for proving robust satisfaction of LTL formulas of discrete time hybrid systems.

3.2. *Under-Approximating Abstraction*

Now we also construct a finite system that under-approximates the original system S, hence $[\![S]\!]$ now abstracts the under-approximation. We first present a simple and succinct approach, which helps to simplify the proof, and then point out possible improvements to achieve practical efficiency.

We choose a sample point $s(p)$ for every predicate $p \in P$. Then let $\underline{\alpha}_P(S)$ be the transition system whose transition relation is the set of all $\langle s(p), s(p')\rangle$ such that $p, p' \in P$, and $\langle s(p), s(p')\rangle \models \tau$, and for which for every $i \in \{0, \ldots, k\}$, the i-th observed proposition contains the set of all $s(p)$ such that $p \in P$, $s(p) \models \pi_i$. In the case without transcendental function symbols this can be effectively constructed, and due to an identity abstraction function $\underline{\alpha}_P(S)$ is abstracted by $[\![S]\!]$, which ensures the correctness of the under-approximation.

Now let, for a constraint ϕ, $\underline{[\![\phi]\!]}_\varepsilon$ be the set of all x for which there is no ϕ^* with $d(\phi, \phi^*) \leq \varepsilon$ and no x^* with $d(x, x^*) \leq \varepsilon$ such that $x^* \not\models \phi^*$, which is the minimal element of the ε-perturbed solution sets of wrt. to the subset relation $\subseteq$. Again we extend this to hybrid systems, and use the result to bound the under-approximation error as follows:

Theorem 18. $\underline{[\![S]\!]}_{diam(P)} \preceq \underline{\alpha}_P(S)$

We do not include a proof here since it can be adapted from the proof of Theorem 19 below.

So, instead of falsifying an LTL formula against the original system S we can check it against $\underline{\alpha}_P(S)$. Moreover, by letting the diameter of P go to zero, this check will eventually succeed for robust systems.

However, there is no known algorithm that can check in all cases whether a term containing constants, addition, multiplication, and transcendental function symbols is zero. The reason is that it is not known how many digits after the comma have to be zero to be able to decide that the constant is zero. Hence, in that case, it also not in general possible to check the transition relation and observed propositions against the sample points.

In our solution we use interval arithmetic to enclose the value of the terms of the constraints at the sample points into small intervals of bounded width (or alternatively, compute their value up to a certain precision), and compute a conservative under-approximation based on that information. Again, this ensures correctness of the under-approximation ($\underline{\alpha}^I_P(S) \preceq [\![S]\!]$). Moreover, since for robustly false problems finite precision suffices for falsification, by making the width of the computed intervals go to zero as the diameter of the abstraction goes to zero, the falsification will eventually succeed:

Theorem 19. *There is a function* $E : \mathbb{R}^+ \to \mathbb{R}^+$ *with* $\lim_{x\to 0} E(x) = 0$*, such that given a set of predicates* P*,* $\underline{[\![S]\!]}_{E(diam(P))} \preceq \underline{\alpha}^I_P(S)$

Proof. Let E_s be the function that assigns to ε the maximal width of the intervals enclosing the terms used for computing $\underline{\alpha}^I_P(S)$. Choose for E the function such that $E(\varepsilon) = \max\{\varepsilon, E_s(\varepsilon)\}$. Assume that $\underline{[\![S]\!]}_{E(diam(P))}$ has the form $(\to, Q_0, Q_1, \ldots, Q_r)$, and $\underline{\alpha}^I_P(S)$ has the form $(\to', Q'_0, Q'_1, \ldots, Q'_r)$.

For proving $\underline{[\![S]\!]}_{E(diam(P))} \preceq \underline{\alpha}^I_P(S)$, we use an abstraction function H such that $H(x)$ is $s(p)$ where p is a predicate in P such that $x \models p$. We prove the two items of Definition 9 as follows:

- Let $i \in \{0, \ldots, r\}$ be arbitrary, but fixed. Let $x \in Q_i$. We have to prove that $H(x)$ is in the corresponding element Q'_i of $\underline{\alpha}^I_P(S)$. By definition of H this means to prove that for all p with $p \in P$ and $x \models p$, $s(p) \in Q'_i$, that is, interval evaluation of π_i on $s(p)$, as used in the construction of $\underline{\alpha}^I_P(S)$, yields a true value.

  Since the distance between x and $s(p)$ is smaller than $diam(P)$ and hence also smaller than $E(diam(P))$, and due to the definition of $\underline{[\![S]\!]}_{E(diam(P))}$ we know that for all constraints π^*_i with $d(\pi_i, \pi^*_i) \leq E(diam(P))$, and hence also for all constraints π^*_i with $d(\pi_i, \pi^*_i) \leq E_s(diam(P))$, $s(p) \models \pi^*_i$. Hence interval evaluation of π_i will be precise enough to ensure that $s(p) \in Q'_i$.
- Let x, x_1 be such that $x \to x_1$. We have to prove that $H(x') \to' H(x_1)'$. By definition of H this means to prove that for all p and p_1 with $p, p_1 \in P$, $x \models p$, and $x_1 \models p_1$, we have that $s(p) \to' s(p_1)$, that is interval evaluation of τ on $\langle s(p), s(p_1)\rangle$, as used in the construction of $\underline{\alpha}^I_P(S)$, yields a true value.

  Since the distance between x and $s(p)$ and the distance between x_1

and $s(p_1)$ are both smaller than $diam(P)$, and hence also smaller than than $E(diam(P))$, and due to the definition of $\underline{[\![S]\!]}_{E(diam(P))}$ we know that for all constraints τ^* with $d(\tau, \tau^*) \leq E(diam(P))$, and hence also for all constraints τ^* with $d(\tau, \tau^*) \leq diam(P)$, $\langle s(p), s(p_1)\rangle \models \tau^*$. Hence interval evaluation of τ will be precise enough to ensure that $s(p) \to' s(p_1)$. □

This approach will typically need a set of predicates with a very small diameter to be able to guarantee the existence of transitions. We now sketch some improvements, based on numerical local search methods and argue, that for realistic designs this will lead to drastic improvements.

We exploit typical testing strategies for dynamic systems. Consider as an example electronic stability control applications. Test engineers would typically force a car into areas where the different modes of such an envelope protection system become active, testing each mode in isolation. For example, they would from a stable driving situation speed into a curve in a way causing oversteering without control intervention, and after recovery of stability proceed to a next maneuver, for example, testing the ability to control partial road-icing. Such examples teach two lessons:

- Test-cases require only a bounded history - each threat of violating envelop protection, as well as the timely invocation of control-strategies compensating for the attack on car stability, can be tested in isolation, starting from a "home state", in which the car is stable.
- Secondly, the test driver is forcing the open control inputs (e.g., angle of steering wheel, rate of acceleration) towards meeting the triggering condition for the "relevant" mode of the envelope protection algorithm. Heuristics for efficient construction of under-approximations should thus find settings of open control inputs moving the plant from a stable "home state" towards predicates guarding transitions towards recovery modes of the system.

This motivates an approach that analyzes sequences of predicates $p_0, \dots, p_l$ whose corresponding transitions have not yet been excluded in an over-approximating abstraction. It then uses using an iterative numerical method with a sample point in each predicate as starting point to find a concrete trajectory running through this sequence. By recursively analyzing longer and longer sequences the method will build longer and longer concrete trajectories. The above analysis shows that this will eventually lead to trajectories from home state to home state, allowing us to build transitions of the underapproximation for which we know that they can be concatenated, and hence be used to form concrete counter-examples.

Since the approach uses sample points as starting points for the numerical search method, it inherits the convergence properties ensured by the theorems above. And since we know from the above analysis that usually an analysis of sequences of short length suffices, the combinatorial blowup of analyzing *all* abstract-counterexamples can be avoided.

3.3. *Proving Robust Satisfaction and Falsification*

Now assume as given a temporal specification $\varphi \in LTL$, with the arithmetic atoms $\Pi = \{\pi_0, \ldots, \pi_n\}$ occurring negatively. Since the aim of the current paper is to establish the overall approach, we only give a basic algorithm for abstraction refinement, whose efficiency can be significantly improved according to the directions outlined below. The key result of this section is, that the abstraction refinement algorithm is guaranteed to terminate, if φ is robustly satisfied or falsified by S.

We introduce the following basic algorithm: Create a sequence of partitions $P_0, P_1, \ldots$ such that the diameter of the partitions goes to zero. If at a certain iteration i, $\underline{\alpha}^I_{P_i}(S)$ falsifies φ, terminate with the result that φ is robustly falsified in S. If $\overline{\alpha}^I_{P_i}(S)$ satisfies φ, terminate with the result that φ is robustly satisfied by S. Here we start with $P_0 = \{\langle m, I^{x_1} \times \ldots \times I^{x_k}\rangle \mid m \in M\}$ as the initial partition. We ensure that the diameter of the partition goes to zero by splitting the largest box in P_m along the biggest side-length to obtain P_{m+1} (recall that the real-valued variables range over bounded intervals).

Before providing our main result and applying the algorithm to a more complex example, we illustrate its behavior on a toy example of a safety verification problem. To keep the example simple, we concentrate on its continuous behavior and completely ignore possible discrete behavior. We assume that the constraint τ is of the form $x_1' = x_1/2 \wedge x_2' = x_2/2$, that the set of initial states is described by a constraint π_0 of the form $x_1 \leq 3 \wedge x_2 \leq 3$ and the set of unsafe states is described by a constraint π_1 of the form $x_1 \geq 6$. The assumption of a state space $[0,8] \times [0,10]$ will lead to the initial partition $\{[0,8] \times [0,10]\}$. The corresponding abstraction consists of the single abstract state $[0,8] \times [0,10]$ with the set of initial states and set of unsafe states both being equal to $\{[0,8] \times [0,10]\}$ (see Figure 2, where the left-hand side shows the partitioning of the state space, and the right-hand the corresponding abstraction). This abstraction is clearly unsafe, and hence a finer partition is needed.

The algorithm will then split $[0,8] \times [0,10]$ along the biggest side-length which will result in an abstraction consisting of the two abstract states $[0,8] \times [0,5]$, $[0,8] \times [5,10]$. This abstraction is still not safe since $[0,8] \times [0,5]$ is still both in the set of initial and the set of unsafe states (see Figure 3).

After two further splittings, the abstraction consisting of the four abstract states $[0,4] \times [0,5]$, $[0,4] \times [5,10]$, $[4,8] \times [0,5]$, and $[4,8] \times [5,10]$ (see Figure 4) will be safe, proving the safety of the original system.

The following main result opens a new line of attack to the verification of non-linear hybrid systems.

Theorem 20. *The basic algorithm is guaranteed to terminate with definite answer if S robustly satisfies φ or φ is robustly falsified by S.*

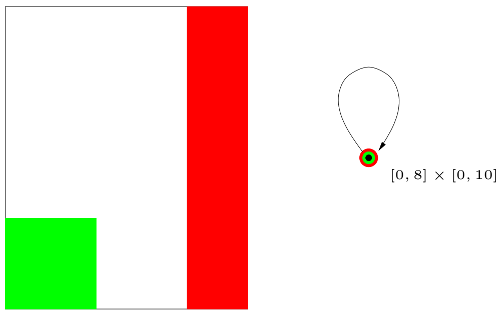

Fig. 2. Initial Abstraction

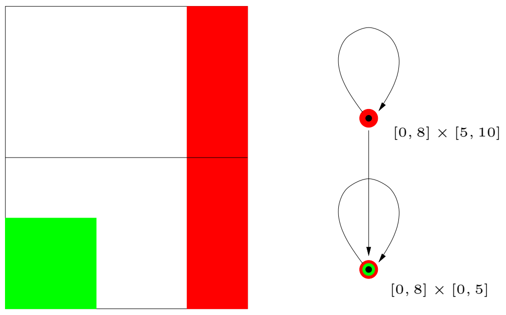

Fig. 3. Refined Abstraction

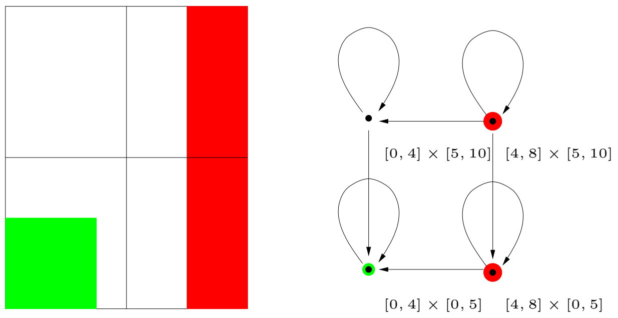

Fig. 4. Safe Abstraction

Proof. The abstraction refinement procedure ensures that the diameter of the abstraction goes to zero. If S robustly satisfies φ, the fact that $\gamma(\overline{\alpha}^I_{P_m}(S))$ abstracts $\overline{\alpha}^I_{P_m}(S)$ due to Theorem 17, and transitivity of abstraction implies that $\overline{\alpha}^I_{P_m}(S)$ is abstracted by $\overline{[\![S]\!]}_\varepsilon$ with ε going to zero as m goes to infinity. Let $r > 0$ be such that $\overline{[\![S]\!]_r} \models \varphi$ which is ensured by robustness. Thus, there is an m, from which on ε will be smaller than r. Then $\overline{\alpha}_{P_m}(S) \models \varphi$ and the algorithm succeeds. The case where φ is robustly falsified in S is similar. □

Note that this theorem also includes robust progress properties. In that case, the algorithm will eventually remove all unnecessary transitions in the abstraction that lead from a predicate to itself.

Clearly, further work is needed, to make this algorithm practically efficient. Still—in order to evaluate its efficiency potential—we have implemented it using the programming language O'Caml. Our implementation keeps a list of boxes for each mode, starting with one box per mode. It computes the abstraction by going through each box, and marking it as initial or unsafe if interval arithmetic cannot disprove the corresponding constraint. Moreover, on every pair consisting of this box and another box, it checks whether interval arithmetic can disprove the transition constraint, and if it cannot, it puts a transition in the abstraction. If this abstraction is safe, verification is finished. If it is not, the box with the biggest side-length is split, and the abstraction is recomputed. This continues until a safe abstraction can be found, which is ensured for robust inputs due to the results of this paper.

For the interval arithmetic checks, the implementation uses the constraint propagation engine RSOLVER [27]. In some cases, this allows us, in a similar way as in the continuous time case [29, 28], to prove that only a part of a given box can be reachable from any other box, in which case we only keep this smaller part in the box list. Moreover, after splitting a box, we do not recompute the whole abstraction, but only the part that was involved in the split.

Already in this basic form, the algorithm yields promising results for realistic examples. More specifically, for our collision-avoidance example, a designer will ensure that the necessary distance of the two planes should not only be kept for this very example, but this should also be the case when the behavior of the planes changes slightly. The results of this paper imply, that in such a robust case our verification algorithm terminates.

In fact, for $\delta = 0.6$, our interval arithmetic prototype proves in about 20 seconds that the safety margin of the planes is maintained, using an abstraction that has only 100 boxes.

4. Conclusion

This paper opens a novel line of attack to the verification of non-linear hybrid systems. We have argued for the naturalness of the notion of robust satisfaction, and demonstrated how to construct a series of increasingly more accurate abstractions, which for robust designs is guaranteed to converge to a sufficiently precise model to prove or falsify temporal specifications of hybrid systems in a rich specification logic with first-order arithmetic constraints, able to express real-time requirements. Though we have chosen LTL as the temporal framework in this paper, the development only exploits safeness of the constructed abstractions; it is well known [13], that also ACTL* properties are preserved under the performed abstractions.

The base algorithm is compatible with many of the optimization techniques for abstraction refinement. Promising directions for optimization currently under investigation in the large scale collaborative research project AVACS include:

Initial Partitioning: We refine B_0 to approximately discriminate all guards and arithmetic constraints in Π, over-approximating their shapes by boxes. This approach is already realized as part of another research activity for verification of hybrid systems based on predicate abstraction techniques [7].

Counterexample guided abstraction refinement: We incrementally analyze counterexample fragments for concretization [11]. We do so, by applying the constraint propagation based solver for non-linear constraints [27] to the corresponding first-order formula. If the constraint is unsolvable, we dismiss the counterexample fragment as spurious by encoding the corresponding information into an automaton-based representation of the abstraction.

Local search for counter-examples: Instead of just testing samples in the abstract states for counter-examples, we use local search (based on a Newton-like method) to find samples that form counter-examples.

We see this paper hence as a promising starting point in exploiting the usage of interval-based constraint solving techniques for the verification of non-linear hybrid systems.

Acknowledgements: The authors are grateful to the anonymous reviewers of the paper for their helpful comments.

References

[1] Manindra Agrawal, Frank Stephan, P S Thiagarajan, and Shaofa Yang. Behavioural approximations for restricted linear differential hybrid automata. In Hespanha and Tiwari [20], pages 4–18.

[2] Manindra Agrawal and P S Thiagarajan. The discrete time behaviour of lazy linear hybrid automata. In Morari and Thiele [23], pages 55–69.

[3] Eugene Asarin, Thao Dang, and Oded Maler. The d/dt tool for verification of hybrid systems. In *CAV'02*, number 2404 in LNCS, pages 365–370. Springer, 2002.

[4] Tom Bienmüller, Jürgen Bohn, Henning Brinkmann, Udo Brockmeyer, Werner Damm, Hardi Hungar, and Peter Jansen. Verification of automotive control units. In Ernst-Rüdiger Olderog and Bernd Steffen, editors, *Correct System Design*, volume 1710 of *LNCS*, pages 319–341. Springer, 1999.

[5] Tom Bienmüller, Udo Brockmeyer, Werner Damm, et al. Formal verification of an avionics application using abstraction and symbolic model checking. In Felix Redmill and Tom Anderson, editors, *Towards System Safety—Proc. of the 7th Safety-critical Systems Symp.*, pages 150–173. Springer, 1999.

[6] Eckard Böde, Werner Damm, Jarl Høyem, Bernhard Josko, Jürgen Niehaus, and Marc Segelken. Adding value to automotive models. In *ASWSD 2004. ARTIST and NSF Workshop on Automotive Software Development*, 2004.

[7] Jürgen Bohn, Werner Damm, Orna Grumberg, et al. First-order-CTL model checking.

In V. Arvind and R. Ramanujam, editors, *Foundations of Software Techn. and Theor. Comp. Sc.*, volume 1530 of *LNCS*, pages 283–294. Springer, 1998.

[8] J.-Y. Brunel, W. Damm, A. Ferrari, U. Freund, B. Josko, S. Kowalewski, A. Sangiovanni-Vincentelli, M. Torngren, T. Thurner, and H. von Hasseln. The future design scenario and the sea initiative. In *IFAC Symposium on Advances in Automotive Control*, University of Salerno, Italy, 2004.

[9] John Canny. Some algebraic and geometric computations in PSPACE. In *STOC '88: Proceedings of the twentieth annual ACM symposium on Theory of Computing*, pages 460–469, New York, NY, USA, 1988. ACM Press.

[10] A. Chutinan and B. H. Krogh. Computing polyhedral approximations to flow pipes for dynamic systems. In *The 37th IEEE Conference on Decision and Control: Session on Synthesis and Verification of Hybrid Control Laws (TM-01)*, 1998.

[11] E. Clarke, A. Fehnker, Z. Han, B. Krogh, J. Ouaknine, O. Stursberg, and M. Theobald. Abstraction and counterexample-guided refinement in model checking of hybrid systems. *Int. Journal of Foundations of Comp. Science*, 14(4):583–604, 2003.

[12] Edmund M. Clarke, Orna Grumberg, and David E. Long. Model checking and abstraction. *ACM Transaction on Programming Languages and Systems*, 16(5):1512–1542, 1994.

[13] Edmund M. Clarke, Orna Grumberg, and Doran A. Peled. *Model Checking*. MIT Press, 1999.

[14] W. Damm, H. Hungar, and E.-R. Olderog. Verification of cooperating traffic agents. *International Journal of Control*, 79(5):395–421, 2006.

[15] Martin Fränzle. Analysis of hybrid systems: An ounce of realism can save an infinity of states. In J. Flum and M. Rodriguez-Artalejo, editors, *Computer Science Logic (CSL'99)*, number 1683 in LNCS. Springer, 1999.

[16] Martin Fränzle. What will be eventually true of polynomial hybrid automata. In N. Kobayashi and B. C. Pierce, editors, *Theoretical Aspects of Computer Software (TACS 2001)*, number 2215 in LNCS. Springer-Verlag, 2001.

[17] Antoine Girard and George Pappas. Verification using simulation. In Hespanha and Tiwari [20], pages 272–286.

[18] Thomas A. Henzinger, Pei-Hsin Ho, and Howard Wong-Toi. Algorithmic analysis of nonlinear hybrid systems. *IEEE Transactions on Automatic Control*, 43:540–554, 1998.

[19] Thomas A. Henzinger, Benjamin Horowitz, Rupak Majumdar, and Howard Wong-Toi. Beyond HyTech: hybrid systems analysis using interval numerical methods. In N. Lynch and B. Krogh, editors, *Proc. HSCC'00*, volume 1790 of *LNCS*. Springer, 2000.

[20] João Hespanha and Ashish Tiwari, editors. *Hybrid Systems: Computation and Control*, volume 3927 of *LNCS*. Springer, 2006.

[21] Luc Jaulin, Michel Kieffer, Olivier Didrit, and Éric Walter. *Applied Interval Analysis, with Examples in Parameter and State Estimation, Robust Control and Robotics*. Springer, Berlin, 2001.

[22] W. S. Levine, editor. *The Control Handbook*. CRC Press and IEEE Press, 1995.

[23] M. Morari and L. Thiele, editors. *Hybrid Systems: Computation and Control*, volume 3414 of *LNCS*. Springer, 2005.

[24] Arnold Neumaier. *Interval Methods for Systems of Equations*. Cambridge Univ. Press, Cambridge, 1990.

[25] T. Peikenkamp, E. Böde, I. Brückner, H. Spenke, M. Bretschneider, and H.-J. Holberg. Model-based safety analysis of a flap control system. In *Proceedings of the*

INCOSE 2004 – 14th Annual International Symposium, Toulouse, 2004.

[26] Stefan Ratschan. Quantified constraints under perturbations. *Journal of Symbolic Computation*, 33(4):493–505, 2002.

[27] Stefan Ratschan. RSolver. *http://rsolver.sourceforge.net*, 2004. Software package.

[28] Stefan Ratschan and Zhikun She. Safety verification of hybrid systems by constraint propagation based abstraction refinement. *ACM Journal in Embedded Computing Systems*. To appear.

[29] Stefan Ratschan and Zhikun She. Safety verification of hybrid systems by constraint propagation based abstraction refinement. In Morari and Thiele [23], pages 573–589.

[30] B. I. Silva and B. H. Krogh. Formal verification of hybrid system using checkmate: A case study. In *American Control Conference*, 2000.

[31] O. Stursberg, S. Kowalewski, and S. Engell. On the generation of timed discrete approximations for continuous systems. *Mathematical and Computer Models of Dynamical Systems*, 6:51–70, 2000.

[32] Paulo Tabuada and George Pappas. Model checking ltl over controllable linear systems is decidable. In *Hybrid Systems: Computation and Control*, number 2623 in LNCS, pages 498–513. Springer-Verlag, 2003.

[33] A. Tarski. *A Decision Method for Elementary Algebra and Geometry*. Univ. of California Press, Berkeley, 1951.

[34] Claire Tomlin, George J. Pappas, , and Shankar Sastry. Conflict resolution for air traffic management: A study in multi-agent hybrid systems. *IEEE Transactions on Automatic Control*, 43(4), 1998.